PCT/GB99/01773

09/674720

GB 99/01773

Europäisches Patentamt | European Patent Office | Office européen des brevets

Bescheinigung | Certificate | Attestation

Die angehefteten Unterlagen stimmen mit der ursprünglich eingereichten Fassung der auf dem nächsten Blatt bezeichneten europäischen Patentanmeldung überein.

The attached documents are exact copies of the European patent application described on the following page, as originally filed.

Les documents fixés à cette attestation sont conformes à la version initialement déposée de la demande de brevet européen spécifiée à la page suivante.

Patentanmeldung Nr. | Patent application No. | Demande de brevet n°

98309609.0

PRIORITY DOCUMENT

SUBMITTED OR TRANSMITTED IN
COMPLIANCE WITH
RULE 17.1(a) OR (b)

Der Präsident des Europäischen Patentamts;
Im Auftrag

For the President of the European Patent Office

Le Président de l'Office européen des brevets
p.o.

Aslette Fiedler

A. Fiedler

EN HA , DEN
E HA E, 07/06/99
HAYE, LE

Europäisches Patentamt | European Patent Office | Office européen des brevets

Blatt 2 der Bescheinigung
Sheet 2 of the certificate
Page 2 de l'attestation

Anmeldung Nr.:
Application no.:
Demande n°: 98309609.0

Anmeldetag:
Date of filing:
Date de dépôt: 24/11/98

Anmelder:
Applicant(s):
Demandeur(s):
BRITISH TELECOMMUNICATIONS public limited company
London EC1A 7AJ
UNITED KINGDOM

Bezeichnung der Erfindung:
Title of the invention:
Titre de l'invention:
Communications network with tariff based on network load

In Anspruch genommene Priorität(en) / Priority(ies) claimed / Priorité(s) revendiquée(s)

Staat: State: Pays:	Tag: Date: Date:	Aktenzeichen: File no. Numéro de dépôt:

Internationale Patentklassifikation:
International Patent classification:
Classification internationale des brevets:
H04L12/14, H04L12/56

Am Anmeldetag benannte Vertragstaaten:
Contracting states designated at date of filing: AT/BE/CH/CY/DE/DK/ES/FI/FR/GB/GR/IE/IT/LI/LU/MC/NL/PT/SE
Etats contractants désignés lors du depôt:

Bemerkungen:
Remarks:
Remarques: **The original title of the application reads as follows:**
Communications network

Communications Network

The present invention relates to a communications network, and in particular to charging mechanisms in such a network. It includes aspects of the inventions disclosed and claimed in the present applicant's co-pending British patent application no. 9812161.9 filed 5 June 1998 and the contents of that earlier application are incorporated herein by reference.

In conventional communications networks, such as national PSTNs (public switched telephone networks), a significant proportion of the network resources are devoted to metering and billing network usage. Studies have estimated these resources as consuming as much as 6% of the revenue of a telecommunications company. The Internet, by contrast, does not in general incorporate metering and billing mechanisms for individual customers. The absence of the network infrastructure required to support metering and billing reduces the operational costs of the Internet compared to conventional telephony networks, and has facilitated the rapid expansion of the Internet . However the absence of appropriate billing mechanisms has significant disadvantages in terms of the characteristics of the traffic carried by the internet: it encourages profligate use of network resources, and diminishes the incentive for investment in network infrastructure to support new applications requiring, e.g., guaranteed quality of service (QoS).

According to a first aspect of the present invention, there is provided a method of operating a communications network, including automatically varying, depending on network loading as detected at a customer terminal, a tariff for network usage by the customer terminal.

The present invention provides an approach to charging for network usage which involves little or no network overheads, and which moreover is able to reflect local variations in network loading. This is achieved by detecting a measure of network loading at the customer terminal. This measure will reflect those resources being used by the customer terminal at a particular time. For example, in a federated data network comprising a number of sub-domains, such as the Internet, a customer terminal in one subdomain may attempt to access a server in another subdomain at a time when there is overloading of a router somewhere in the path from the customer terminal to the server. Then, even if average loading of the network as a whole is

low, the loading as perceived by the customer terminal is high, and a tariff for use of the network by the customer terminal is increased to reflect the scarcity of the relevant network resources. Similarly other terminals attempting to access data via the same router will also perceive the network loading as being high, and will also have their tariffs increased accordingly. With an appropriately steep rise in tariff, at least some customer terminals will then choose to abandon or defer their attempts to use the network resource in question, thereby relieving the loading of the router.

This aspect of the invention may be used in systems in which the end user is merely informed of the price and accounting and billing is carried out by the network provider, also in systems where the end user measures their usage and supplies this information to the network provider, and also in systems where the end user both measures their usage and calculates the payment due to the network provider.

The method may include detecting at the customer terminal a network performance parameter which depends on network loading, and varying the tariff depending on the network performance parameter. When the network is a packet network, then the network performance parameter may be the number of packets lost in transmission between a data source and the customer terminal. This approach has the advantage that it is suitable for implementation using existing congestion control mechanisms in packet networks. For example, in the Internet, routers respond to congestion by dropping packets, and a customer terminal using TCP (Transport Control Protocol) detects packet loss and controls a congestion window at the terminal accordingly. The present invention may use the detected packet loss to trigger an increase in the tariff applicable to the terminal. This may be accompanied by the signalling of congestion by the receiving terminal to the data source.

In an alternative and preferred approach, when congestion is detected within the network an explicit congestion signal may be generated and transmitted to the customer terminal, and the receipt of the explicit congestion signal at the customer terminal may then trigger the increase in the tariff. This approach has the advantage that an appropriate congestion signal may be generated in advance of significant degradation in network performance occurring, making it easier to maintain a required quality of service. Preferably the explicit congestion signal is carried with a data

packet on the communications network. Preferably a router in the network writes an explicit congestion signal in a packet when congestion is detected at the router.

As a further alternative, the congestion notification signal may be a source quench signal generated at the router and transmitted back to the data source.

Preferably there is a non-linear relationship between between the increase in tariff and the detected network loading. Preferably the method includes making a first relatively smaller increase in the tariff when congestion is first detected, and making at least one further, relatively larger increase, if the congestion persists. This behaviour may be pre-programmed into the tariff algorithm. In this way oscillatory behaviour of the network can be avoided, while ensuring that a sufficient increase is made to have a desired impact on the demand for network resources.

Preferably the method includes programming a decision agent at the customer terminal with user-determined price criteria, and comparing a price calculated using the tariff with the said price criteria. For example, the decision agent might be programmed to proceed with data communications as long as a per-packet charge is zero or is less than a predetermined threshold. Then if congestion causes the charge to rise above that threshold, the data communication is interrupted and the decision agent modifies the operation of the customer terminal. For example the decision agent may slow down communications once the cost threshold is exceeded. The user may then choose to continue with the communication, or may elect to abandon the attempt to access data from a particular source. The decision agent may be programmed with rules which relate to an overall price for multiple simultaneous transmissions and which accord different weights to different respective applications.

Preferably the method includes distributing a tariff algorithm via the communications network to a plurality of terminals and calculating at each terminal using the tariff a charge for network usage by the terminal. In this case preferably the method includes steps, carried out by a network operator, of ntermittently sampling traffic between a customer terminal and the network, and as part of the sampling recording network loading affecting the customer terminal;

for the sampled traffic comparing a charge calculated by the customer terminal and an expected charge and detecting thereby any discrepancy.

The step of recording network loading may involve, for example, recording packet drops, or explicit congestion notification (ECN) bits or source quench messages.

The invention also encompasses a network operating by a method in accordance with the first aspect of the invention, and customer terminals, network routers and other network nodes for use in such a network.

Systems embodying the present invention will now be described in further detail, by way of example only, with reference to the accompanying drawings, in which:

Figure 1 is a schematic showing a network embodying the invention;

Figures 2a and 2b are graphs showing tariff functions;

Figure 3 shows the format of a differential service byte;

Figure 4 is a schematic showing the component objects of a charging architecture for use with the network of Figure 1;

Figure 5 shows data passed between the accounting objects of Figure 4;

Figure 6 is a schematic showing protocol stacks on a customer terminal and in the network domain;

Figure 7 is a graph showing the variation of tariff with time.

As shown in Figure 1, a communications network 1 includes a number of network sub-domains 2A-C. The network sub-domains may be under the control of different operators who may not trust each other. The network subdomains are interconnected by gateway routers 3, 4. In the present example the communications network is the Internet and supports both unicast and multicast Internet Protocol (IP) and associated protocols. A customer terminal 5 is connected via a public switched telephony network (PSTN) 6 and an access router 7 to a subdomain 2A. A single blocking test is applied to traffic at this point of access. The gateway routers 3,4, and access router 7 may be commercially available devices such as CISCO series 7500 routers and CISCO series AS5800 universal access server respectively. Other customer terminals are connected to the network, including a Java-enabled mobile terminal 8 and a data server 9.

In addition to the local tariff variation mechanism that is described below, the network also uses network-based control of a number of tariff bands. A

network management platform 10 is connected to each subdomain. Each network management platform may comprise, for example, a computing system comprising a SPARC workstation running UNIX (Solaris) together with network management applications. The network management platform 10 hosts management entities and tariff entities. The network management platform communicates with agents 100 in managed devices connected to the respective subdomain, for example using SNMP (simple network management protocol). The management platforms monitors the overall loading of network resources in the respective subdomains, and, as will be further described below, adjust the tariffs for network use accordingly. The Net management platform (NMP) instructs the agent to monitor the device and report aggregated results at regular intervals back to the NMP, so the NMP can monitor the combination of all reports.

Tariff data is communicated to peer tariff entities in other subdomains and also to the customer terminals. The tariff data is multicast using, for example Distance Vector Multicast Routing Protocol (DVMRP) or Protocol Independent Multicast (PIM) dense mode. The tariff data channels are announced and monitored using protocols based on SDP (Session Description Protocol), SAP (Session Announcement Protocol) Charging is carried out on a "pay and display" model. Each customer terminal monitors its own network usage, for example by counting the number of packets it sends or receives across the network interface. It calculates, using a tariff received via the network, the payment due to the network operator, and makes a corresponding payment into an account at the network operator. The network operator polices the use made by customers of the terminal by intermittently sampling traffic to or from a particular customer and comparing the use made and the use paid for.

While centralised monitoring and control of tariffs by the network management platform is effective to respond to global changes in the loading of the network, it is difficult to handle localised congestion in this way. It is difficult to cause a price rise signal to be multicast in such a way that the signal is only received by those attempting to communicate packets through the point of congestion. This would require a separate multicast transmission for each element in the Internet , e.g. a multicast for every different queue on every interface of every router. Alternatively

some aggregation of price rises triggered by local resource loading might be used. This however would mean that price rise signals were sent to users who were not making use of the congested resource. This in turn would make it necessary for the price rise signal to be damped, reducing the ability of the price rise to reduce the demand on the congested resource.

To overcome these difficulties, the tariff algorithms installed on the customer terminals are arranged to respond automatically to congestion in a network resource being used by the terminal. Each algorithm includes a function which varies the price for network usage in dependence upon the detected congestion level. This function may be integrated in the main tariff algorithm, or, as in the example described here may be a separate algorithm used to calculate a premium to be added to a price calculated in accordance with the main tariff algorithm.

The main tariff algorithm calculates a price P as a function of a number of quality parameters, Q_1, Q_2, Q_3 where, for example, Q_1 is a specified latency for packets communicated across the interface between the customer terminal and the network, Q_2 is the reserved bandwidth for the transmission, Q_3 is a specified level of reliability corresponding to a maximum permissible level of packet loss.

The price P is then given by:

$$P = f(Q_1, Q_2, Q_3.....)$$

An example of the pricing function in terms of one of the quality parameters Q is shown schematically in Figure 2a.

The congestion tariff algorithm calculates a premium ΔP which is a function of one or more congestion parameters C:

$$\Delta P = f(C_1, C_2...)$$

The congestion parameters provide a measure of the loading of the resources which a customer terminal is making use of at any given time. In the present example the ratio of packets lost to packets received is used as the congestion parameter. This parameter is readily calculated, for example in the case of packets using TCP (transport control protocol), or RTP (real time protocol) over UDP (user datagram protocol), since such packets include a sequence number. Figure 2b shows one example of the function for generating the premium. In this case, the premium increases as an approximately exponential function of the congestion, so that at low

congestion levels a small premium is charged, while if congestion increases still further, then at higher levels of congestion the premium increases sharply.

In an alternative implementation, an explicit congestion signal is added by any congested router within the network to packets transmitted to the customer terminal.

Although only a single main tariff and premium are described here, in practice different subdomains, and different service providers associated with each subdomain, may each have a different pricing structure, with different main and premium tariffs. However, there is across all the subdomains a common relationship between network loading levels and congestion signalling.

The operation of this second implementation will now be described in the context of a network operating using a differentiated service as described in the Internet Engineering Task Force draft "Differentiated Services Operational Model and Definitions" and in the paper by David D Clark (MIT), "A Model for Cost Allocation and Pricing in the Internet", presented at MIT Workshop on Internet Economics, Mar 1995, URL:http://www.press.umich.edu/jep/works/ClarkModel.html. In a network implementing differentiated services, nodes are arranged to discriminate between packets to provide different levels of service. This capability might be used, for example, to accord delay-sensitive data, such as data generated by an IP telephony client, a higher priority compared to other data, such as email data. At the network edge, for example at a client terminal running the IP telephony client, bits in a TOS (type of service) octet contained within each packet header are set to indicate the appropriate service level. Those bits are used by routers within the network to determine how the relevant packets should be handled.

The TOS octet when used in this way is termed the DS (differential service) byte. The format of the differential service byte is shown in Figure 3. Bit zero, labelled "IN" indicates whether the packet is inside or outside a defined profile. Bits 1 to 5 labelled "PHB" define a "per-hop-behaviour" that is they indicate how, for example, a router should handle the packet, e.g. by according it lower or higher priority. Bits 6 to 7, in this particular form of the DS byte, are used for explicit congestion notification (ECN). One of these bits is set to indicate whether the routers in the path of the packet are capable of setting the ECN field, and the other

bit is used as a flag which is set (by ECN capable routers) when congestion, or loading which would potentially lead to congestion, occurs. Random Early Detection (RED) algorithms are currently implemented in routers. These algorithms measure average queue length within the packet buffers of a router. An exponential moving average is calculated. When that average queue length exceeds a predetermined threshold, then the router signals that congestion is occurring. Conventionally this signalling has been done simply by dropping a packet. However, in the context of an ECN scheme, the router, instead of dropping a packet, sets an ECN bit in a packet header to indicate that congestion is occurring. This is done probabilistically: that is, some only of the packets passing through the router are marked. The probability of a packet being marked increases with the average queue size. In the rare case that the queue increases to a length where the router buffers are full, then packets are dropped, rather than an ECN bit being set. In this case ECN bits are set for all the remaining packets.

In operation, if the client terminal 5 is accessing a data source on the server 9, congestion may occur, for example, at router 4 which links network sub-domains 2B and 2C. RED-like algorithms in the router 4 detect that the queue lengths in the router buffers, as calculated using the exponential moving average, exceed a predetermined threshold. Accordingly some of the packets from the server 9 en route to the client terminal have the ECN bit of the DS byte set by the router 9 to mark the fact that congestion is occurring. At the client terminal, the DS byte in the headers of incoming packets is read. A moving average of the number of packets containing an ECN bit which is marked is calculated. This average then provides the congestion parameter C_1 which is used to calculate the premium:

$$\Delta P = f(C_1).$$

The total price to the user P_{TOT} is then calculated by adding together the prices determined by main tariff algorithm and by the premium algorithm:

$$P_{TOT} = P + \Delta P.$$

This total price is passed to a cost decision agent running on the client terminal. This cost decision agent is programmed with user defined rules. These might state, for example, that the cost decision agent should authorise the system to proceed with a connection as long as the total cost averaged over a certain time period falls below a

predetermined threshold, e.g. of £0.01 per minute, and that the cost decision agent should suspend a connection and alert the user if the cost rises above that threshold. Alternatively, as previously noted, the cost decision agent may handle several applications simultaneously, and may be programmed to slow down one of the applications as the premium for using a data source accessed by that application increases.

For ease of description, the preceding sections have treated in isolation the local variations in tariff in response to congestion. In practice, this mechanism will in general be combined with other responses to congestion, and with other sources of variation in the tariff. Also, a decision to proceed with a transmission despite congestion will in general require the consent of parties at both ends of the transmission. Considering the entire system of the data source, network and routers and the data receiver, the implementation of an increase in tariff (also termed here a "fine") in response to locally detected congestion occurs as a last resort. Other responses are implemented first, in the following numerical order:

1.the network re-routes around congestion

2.the network borrows capacity from lower levels of service (lower in the context of the relevant dimension(s) of QoS) including the best effort service

3.the network introduces extra capacity (possibly automatically)

4.the end-system establishes that the congestion is on the shared network and not just on the access links or end systems

5.the end-system sets QoS requirements to a "higher" level (if cheaper than the fine for ignoring congestion at the current level)

6.the end-system decides it is essential to ignore the congestion, given the fine for doing

so might be quite high

7.both (all) end-systems agree to ignore the congestion.

Typically, it is at step 4 that an ECN signal is generated. Steps 1 to 3 precede the generation of this signal and steps 5 to 7 follow the generation of the ECN signal.

The last step prior to proceeding with a connection and paying the premium for doing so is establishing agreement by both parties. Accordingly, when the

customer terminal detects congestion, either through receiving explicit congestion notification, or through detection of a relevant parameter such as packet loss, the customer terminal signals this fact back to the or each other end system. In the present example therefore, the client terminal 5 signals to the data server 9 that congestion is occurring. The data server is programmed with rules, which as at the customer may be implemented as an agent, which determine the response to such a signal. For example, the server may refuse service in these conditions. As described previously with respect to Figure 1, in the present example tariffs are multicast through the network from network operators to the customer terminals, and charging is carried out using a "pay and display" process. Figures 4a and 4b shows the objects used to implement the charging architecture in this case. Figure 4a shows the higher level objects and 4b shows the component objects used in a software implementation of the architecture of Figure 4b. In Figure 4a, objects on the client terminal are shown in the half of the Figure labelled "customer" and objects on the access router 7 and the corresponding network sub-domain are shown in the half of the Figure labelled "edge network". The objects on the customer terminal include a session control object S, a customer business rules object B_c , a customer pricing object Pr_c, a QoS manager Q, a customer accounting object Act_c and a customer measurement object M_c. The business rules object B_c receives information on those aspects of the session which involve liability for payment and receives current pricing data from the pricing object Pr_c. The customer business object makes decisions, under the customer's policy control on which chargeable services are utilised, and how much of the chargeable services are utilised. These decisions are fed to the QoS manager Q, which decides which mechanisms are used to achieve the requirements. The QoS manager then controls the customer measurement object M_c to determine which aspects of traffic and service to measure and which aspects to ignore. The measurement object then records the selected aspects of the traffic, for example counting the number of packets received by the customer terminal and the QoS levels for those packets. These data together with the current tariffs, including any premium for congestion, are then used by the customer terminal to determine the charge payable to the network operator. The measurement object M_c is also programmed with instructions which determine the frequency with which it passes

data to the customer accounting object Act_c . The customer accounting object Act_c passes payments to an accounting object Act_p in the network provider's domain.

The accounting objects on the customer terminal may be implemented using a small encrypted flat-file database. On the network provider's side, the equivalent objects may be implemented using a larger database that is scaleable to handle e.g., tens of thousands of customer accounts. An object request broker (ORB) is used for communication between the customer-side objects and the network-side objects, implemented using commercially available tools such as ORBIX (TradeMark) from Iona Technologies plc.

On the network provider's side, that is to say within the subdomain to which the customer terminal is connected, the customer's traffic is measured by a version of M, denoted M_p, but only on a sampling basis determined by the policing function, Po. That is to say, the network operator samples the customer's traffic only intermittently. Po controls where in the network measurements are made in order to capture all of any particular customer's traffic. A bulk measurement function, M_b, is responsible for reporting aggregate traffic levels, as reflected in the moving average of the router queue lengths, to the pricing object, Pr_p. Bulk measurements would typically be collected from across the provider's domain to a centralised pricing function (which would be replicated for reliability). Pr_p sets prices taking into account the business rules from the network provider's business object, B_p, as well as the current traffic levels reported by Mb and pricing from neighbouring providers (see below). The policing function, Po, compares sample measurements from M_p with accounting messages received at Act_p as a result of the customers own measurements . If it establishes that the accounts are insufficient it might restrict service at the access control gateway, Acs, or initiate some other punishment. Encapsulated within the accounting object another policing object checks the accounts match the payments within the contracted time for payment. Finally, the identity mapping function, I, provides a mapping between a customer's identity (account, digital signature, etc.) and their current network address (typically allocated by the ISP, whether unicast or multicast).

Figure 5 shows the data which are passed between the accounting objects. In this example the account data comprises: account identity; bill record identity;

service type identifier; source address; destination address; tariff identity; time; period (i.e. the period covered by the bill record); units; cost; and currency. In addition, the payment data comprises the amount of money and the currency of payment.

Figure 6 shows the measurement region within protocol stacks on the customer terminal and in the network domain. Ideally there would be two measurement points within this region, one trusted by the customer and one trusted by the network, for example at the two points referenced (a) in the Figure. For ease of implementation, a single measurement point (b) trusted by both parties may be used. This might be implemented, for example within a secure module such as a cryptographic card on the client terminal. As an alternative, measurements may be made at different points with some possibility of discrepancies between measurements. On the network the practical measurement point is at the first access device(s) that, for each customer, inspects network layer headers (c)(IP in this case). ISPs should not measure any deeper into their network (d) because their access network and systems will introduce delays and losses.

For an individual customer (e.g. on dial-up access), a practical point at which to measure would also be alongside the network layer but in their end-system's stack (e). Ideally these measurement points would be lower in each stack to be closer to the interface between the two parties and less likely to be affected by contention in the stack. However, measuring at the link layer (f-f) would be inappropriate because only some chargeable parameters set at the network layer will ever be reflected in link layer frames; network level multicast, end-end latency requirements etc. may never be visible at the link layer. Also, link layer headers would need to be ignored when measuring packet sizes for bandwidth calculations to avoid apparent discrepancies where different link technologies are chained together.

In the reception direction (up the stack) this choice of measurement points implies that the lower layers must be dimensioned (buffer sizes, interrupt and thread scheduling priorities) to cope with the most stringent QoS requirements of higher layers. As frames are taken off the physical media, the machine must be able to pass data up the stack without any chance that usage-charged data gets discarded (e.g. due to buffer overflow caused by interrupt contention) before it gets to the network

layer. It is at the network layer where the ISP's service is to be measured and where it is most convenient for QoS requirements to control correct differential treatment of the various flows as they are passed further up the stack (on end-systems) or forwarded (on routers).

The measurement objects described above may be implemented using, with appropriate modifications, publicly available network metering software such as Nevil Brownlee's NeTraMet system. This is a software meter which conforms to the IETF internet accounting architecture described in RFC 2063 and RFC 2064. The meter builds up, using "packet sniffing", packet and byte counts for traffic flows, which are defined by their end-point addresses. Although generally, Addresses can be ethernet addresses, protocol addresses (IP, DECnet, EtherTalk, IPX or CLNS) or 'transport' addresses (IP port numbers, etc), or any combination of these, in the present implementation IP addresses only are used. The traffic flows to be observed are specified by a set of rules, which are downloaded to NeTraMet by a 'manager' program. Traffic flow data is collected via SNMP (Simple Network Management Protocol) from a 'collector' program.

Figure 7 shows how the main tariff determined by the network operator varies in time. In the Figure, curve A is the spot price calculated to reflect the loading of the network at any instant. Curve B is one of a number of different tariff bands. Different tariff bands have different volatilities, and the customer pays a premium for bands offering greater stability. Tariffs are communicated to the customer terminals using a hierarchy of channels carried by the network. An initial contract between a customer and a service provider a single channel address that might typically hold new announcements distributed some months apart (e.g. for contract variations or for new services specifying which second level channel to listen to for tariffs or for downloading new code to handle new tariff structures). The second level channels might deliver updates hours apart which simply announce the addresses of third level channels for the most volatile information. These third level channels may carry updates at intervals of less than a second. Prices for many services may be carried on one channel. For greatest efficiency, this one channel may be split into several channels at times of highest volatility, and re-aggregated into a single channel in more stable periods.

Tables 1 to 7 below list Java source code used to implement two different tariffs. The code of table 1 establishes the operations used for communication between a customer system and a tariff algorithm downloaded by the customer system. Table 2 shows a linear tariff algorithm, in which the tariff depends on the total of the packets sent and packets received by the customer together with a congestion paramater. Table 3 shows the code for generating the customer display in this case. Table 4 shows the code used to display the tariff at the network operator's server. Table 5 shows an exponential tariff algorithm. Table 6 generates the customer display and Table 7 the operator display for the exponential tariff algorithm. By downloading Java code to generate the user interface, that interface can be tailored to the requirements of the particular tariff, and can be adapted as the tariff changes.

TABLE 1

```
import com.sun.java.swing.JComponent;

/**This establishes the operations used for
communication between the customer system
and the downloaded algorithm.
@author Mike Rizzo*/
public interface Tariff {
  JComponent getGUI();
  float getPrice(int pkin, int pkout, int cng);
```

TABLE 2

```
package algorithms.linear;
import com.sun.java.swing.JComponent;
import com.sun.java.swing.JTextField;

import com.bt.jungle.lsma.charging.pricing.Tariff;

public class LinearAlgorithm implements Tariff {
  private float rate;
  private LinearAlgorithmDisplay display;
  public LinearAlgorithm() {
    display = new LinearAlgorithmDisplay();
    setRate(new Float(1));
  }
  public float getPrice(int pkin, int pkout, int cng) {
    return (pkin + pkout + cng) * rate;
  }
  public JComponent getGUI() { return display; }
  public void setRate(Float f) {
    rate = f.floatValue();
    display.setRate(rate);
  }
}
```

```
// Generated by Together
package algorithms.linear;

import com.sun.java.swing.JPanel;
import com.sun.java.swing.JTextField;
import com.sun.java.swing.Box;
import com.sun.java.swing.JLabel;

public class LinearAlgorithmDisplay extends JPanel {
  private JTextField tfRate = new JTextField(4);
  public LinearAlgorithmDisplay() {

    Box vbox = Box.createVerticalBox();

    Box hbox = Box.createHorizontalBox();
    hbox.add(Box.createHorizontalGlue());
    hbox.add(new JLabel("Rate:"));
    hbox.add(Box.createHorizontalGlue());
    hbox.add(tfRate);
    tfRate.setEditable(false);
    hbox.add(Box.createHorizontalGlue());
    vbox.add(hbox);

    add(vbox);
  }
  public void setRate(float f) {
    tfRate.setText(String.valueOf(f));
  }
}
```

TABLE 3

Customer display for linear algorithm

TABLE 4

```
//Generated by Together
package algorithms.linear;

import com.sun.java.swing.*;
import java.awt.event.*;

import com.bt.jungle.lsma.charging.pricing.provider.*;
import com.bt.jungle.util.*;

public class LinearAlgorithmGUI extends JPanel {
  private JTextField tfRate = new JTextField();
  private TuningMessageListener tuningMessageListener;
  private final static String DEFAULT_RATE = "1.0";
  public LinearAlgorithmGUI(TuningMessageListener tml) {
    tuningMessageListener = tml;
    tfRate.setText(DEFAULT_RATE);

    Box vbox = Box.createVerticalBox();

    Box hbox = Box.createHorizontalBox();
    hbox.add(Box.createHorizontalGlue());
    hbox.add(new JLabel("Rate:"));
    hbox.add(Box.createHorizontalGlue());
    hbox.add(tfRate);
    hbox.add(Box.createHorizontalGlue());
    vbox.add(hbox);

    JButton bTransmit = new JButton("Transmit");
    bTransmit.addActionListener(
      new ActionListener() {
        public void actionPerformed(ActionEvent e) {
          transmit();
        }
      }
    );
    hbox = Box.createHorizontalBox();
    hbox.add(Box.createHorizontalGlue());
    hbox.add(bTransmit);
    hbox.add(Box.createHorizontalGlue());
    vbox.add(hbox);

    add(vbox);
  }
  void transmit() {
    try {
      Float f = new Float(tfRate.getText());
      Object args[] = new Object[1];
      args[0] = f;
      tuningMessageListener.notify(
        new Invocation("setRate", args)
      );
    }
    catch (Exception e) {
      e.printStackTrace();
    }
  }
}
```

Provider display for Linear Algorithm

TABLE 5

```
ackage algorithms.exp;
mport com.sun.java.swing.JComponent;
mport com.sun.java.swing.JTextField;

mport com.bt.jungle.lsma.charging.pricing.Tariff;

ublic class ExpAlgorithm implements Tariff {
 private float min;
 private float base;
 private float divisor;
 private ExpAlgorithmDisplay display;
 public ExpAlgorithm() {
   display = new ExpAlgorithmDisplay();
   setMin(new Float(1));
   setBase(new Float(2));
   setDivisor(new Float(10));
 }
 public float getPrice(int pkin, int pkout, int cng) {
   return min + (float) Math.pow(base, (pkin + pkout + cng) / divisor);
 }
 public JComponent getGUI() { return display; }
 public void setMin(Float f) {
   min = f.floatValue();
   display.setMin(min);
 }
 public void setBase(Float f) {
   se = f.floatValue();
   isplay.setBase(base);
 }
 public void setDivisor(Float f) {
   divisor = f.floatValue();
   display.setDivisor(divisor);
 }
```

TABLE 6

```
//Generated by Together
package algorithms.exp;

import java.awt.GridLayout;
import com.sun.java.swing.JPanel;
import com.sun.java.swing.JTextField;
import com.sun.java.swing.Box;
import com.sun.java.swing.JLabel;

public class ExpAlgorithmDisplay extends JPanel {
  private JLabel tfDisplay = new JLabel();
  private float min, base, div;

  public ExpAlgorithmDisplay() {

    add(tfDisplay);
//    tfDisplay.setEditable(false);
    updateDisplay();
  }
  private void updateDisplay() {
    tfDisplay.setText("price = " + min + " + " + base + "^((pkin+pkout+cng)/" + div + ")");
  }
  public void setMin(float f) {
    min = f;
    updateDisplay();
  }
  public void setBase(float f) {
    base = f;
    updateDisplay();
  }
  public void setDivisor(float f) {
    div = f;
    updateDisplay();
  }
}
```

Custom display for exponential algorithms.

TABLE 4

```
// Generated by Together
package algorithms.exp;

import java.awt.GridLayout;
import com.sun.java.swing.*;
import java.awt.event.*;

import com.bt.jungle.lsma.charging.pricing.provider.*;
import com.bt.jungle.util.*;

public class ExpAlgorithmGUI extends JPanel {
 private JTextField tfMin = new JTextField();
 private JTextField tfBase = new JTextField();
 private JTextField tfDivisor = new JTextField();

 private TuningMessageListener tuningMessageListener;
 private final static String DEFAULT_MIN = "1.0";
 private final static String DEFAULT_BASE = "2.0";
 private final static String DEFAULT_DIV = "10.0";

 public ExpAlgorithmGUI(TuningMessageListener tml) {
   tuningMessageListener = tml;
   tfMin.setText(DEFAULT_MIN);
   tfBase.setText(DEFAULT_BASE);
   tfDivisor.setText(DEFAULT_DIV);

   Box vbox = Box.createVerticalBox();

   vbox.add(new JLabel("price = min + pow(base, (pkin+pkout+cng)/divisor)"));

   vbox.add(Box.createVerticalGlue());

   JPanel panel = new JPanel(new GridLayout(3,2));
   panel.add(new JLabel("Minimum"));
   panel.add(tfMin);
   tfMin.addActionListener(
     new ActionListener() {
       public void actionPerformed(ActionEvent e) {
         transmit("setMin", tfMin);
       }
     }
   );
   panel.add(new JLabel("Base"));
   panel.add(tfBase);
   tfBase.addActionListener(
     new ActionListener() {
       public void actionPerformed(ActionEvent e) {
         transmit("setBase", tfBase);
       }
     }
   );
   panel.add(new JLabel("Divisor"));
   panel.add(tfDivisor);
   tfDivisor.addActionListener(
     new ActionListener() {
       public void actionPerformed(ActionEvent e) {
         transmit("setDivisor", tfDivisor);
       }
     }
   );

   vbox.add(panel);

   add(vbox);
 }
 void transmit(String m, JTextField tf) {
   try {
     Float f = new Float(tf.getText());
     Object args[] = new Object[1];
     args[0] = f;
     tuningMessageListener.notify(
       new Invocation(m, args)
     );
   }
   catch(Exception e) {
```

Provider display for Exponential alg.

CLAIMS

1. A method of operating a communications network, including automatically varying, depending on network loading as detected at a customer terminal, a tariff for network usage by the customer terminal.

2. A method according to claim 1, including detecting at the customer terminal a network performance parameter which depends on network loading, and varying the tariff depending on the network performance parameter.

3 A method according to claim 2, in which the network is a packet network and the network performance parameter is the number of packets lost in transmission between a data source and the customer terminal.

4. A method according to claim 1, including detecting a congestion signal at the customer terminal and varying the tariff in response to the congestion signal.

5. A method according to claim 4, including reading a congestion signal at the customer terminal from a data packet received at the customer terminal.

6. A method according to claim 4 or 5, including generating a congestion signal at a router in the network in response to the detection of congestion at the router.

7. A method according to any one of the preceding claims, including making a first relatively smaller increase in the tariff when congestion is first detected, and making at least one further, relatively larger increase, if the congestion persists

8. A method according to any one of the preceding claims, including programming a decision agent at the customer terminal with user-determined price criteria, and comparing a price calculated using the tariff with the said price criteria.

9. A method according to any one of the preceding claims, including distributing a tariff algorithm via the communications network to a plurality of terminals and calculating at each terminal, using the tariff, a charge for network usage by the terminal.

10. A method according to claim 9, further comprising steps, carried out by the network operator, of:

intermittently sampling traffic between a customer terminal and the network, and as part of the sampling recording network loading affecting the customer terminal; and

for the sampled traffic comparing a charge calculated by the customer terminal and an expected charge and detecting thereby any discrepancy.

11. A method according to any one of the preceding claims, in which when the customer terminal detects congestion in data transmitted to the customer terminal from a data source via the network, the customer terminal returns a congestion notification signal to the data source.

12. A communications network including :

means for detecting network loading locally at a customer terminal; and

means responsive to the said means for detecting arranged automatically to vary a tariff for network usage by the customer terminal.

13. A customer terminal for use in a communications network, the customer terminal including:

means for detecting loading of a network to which, in use, the customer terminal is connected;

means responsive to the said means for detecting and arranged automatically to vary a tariff for network usage by the customer terminal.

14. A customer terminal for use in a communications network, the customer terminal including one or more processors arranged to carry out the following steps in sequence:

detecting loading of resources in a network to which the customer terminal is connected; and

automatically varying in dependence on the detected loading a tariff for network useage by the customer terminal.

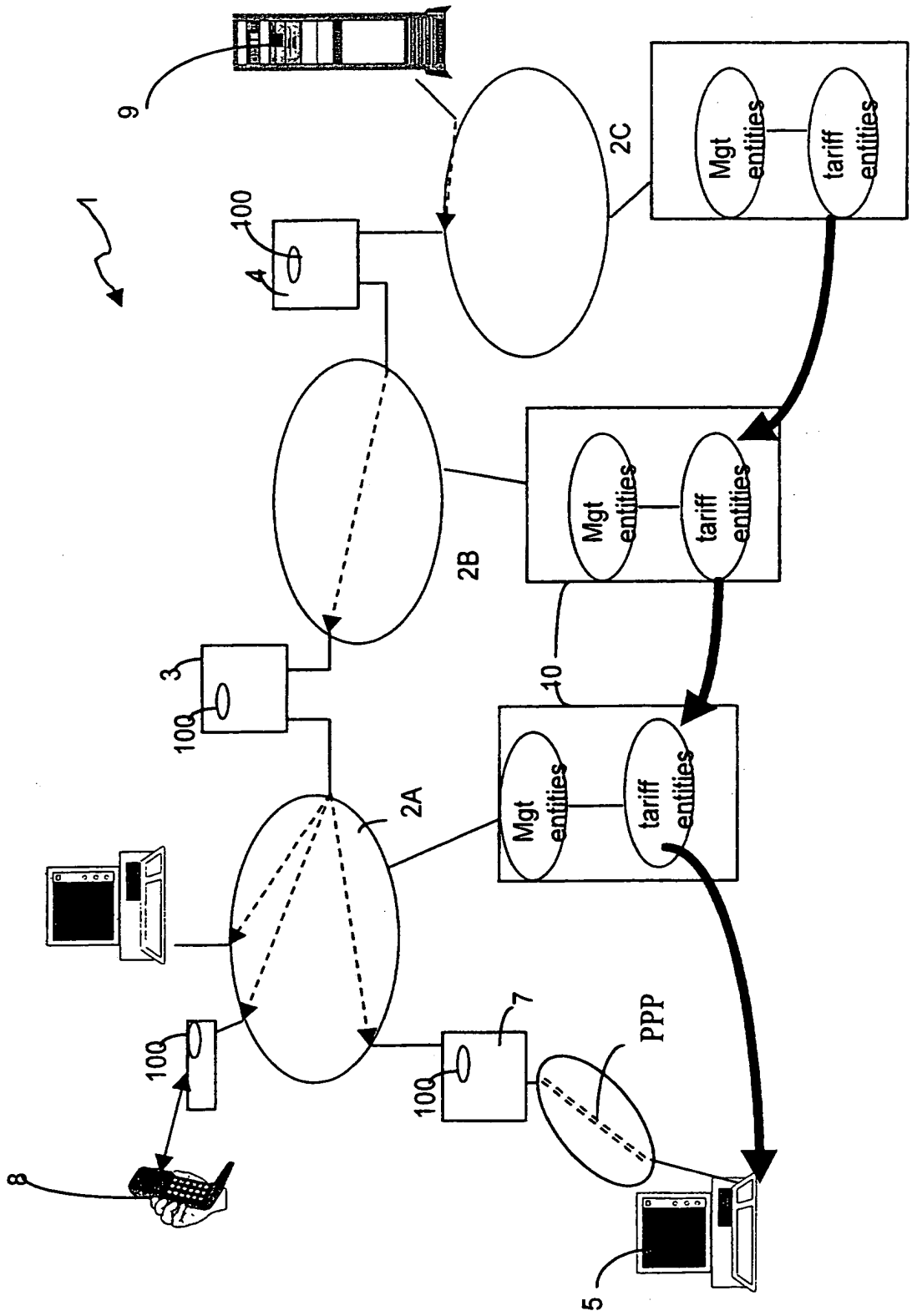

Figure 1

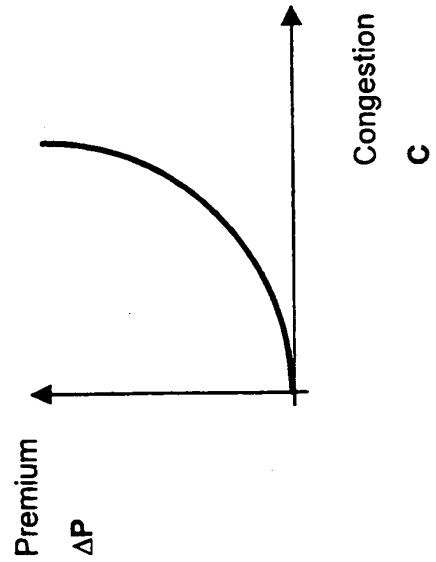

Figure 2b

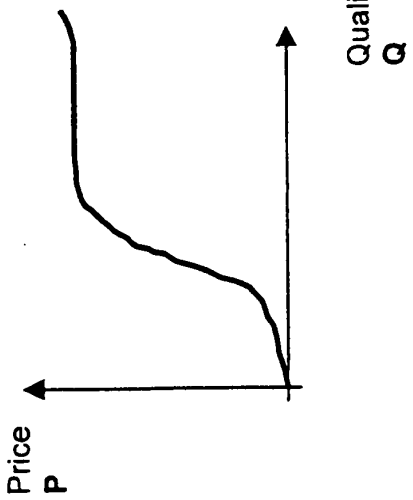

Figure 2a

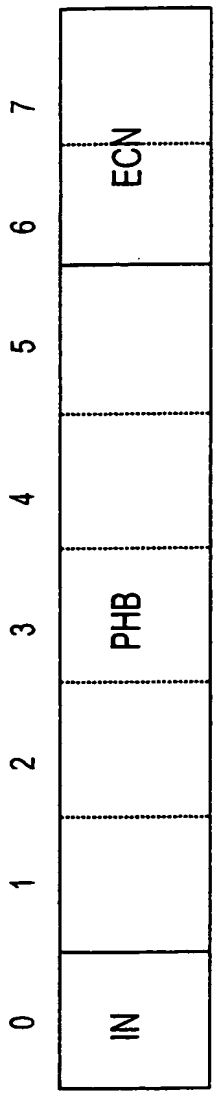

Figure 3

Figure 4a

Figure 4b

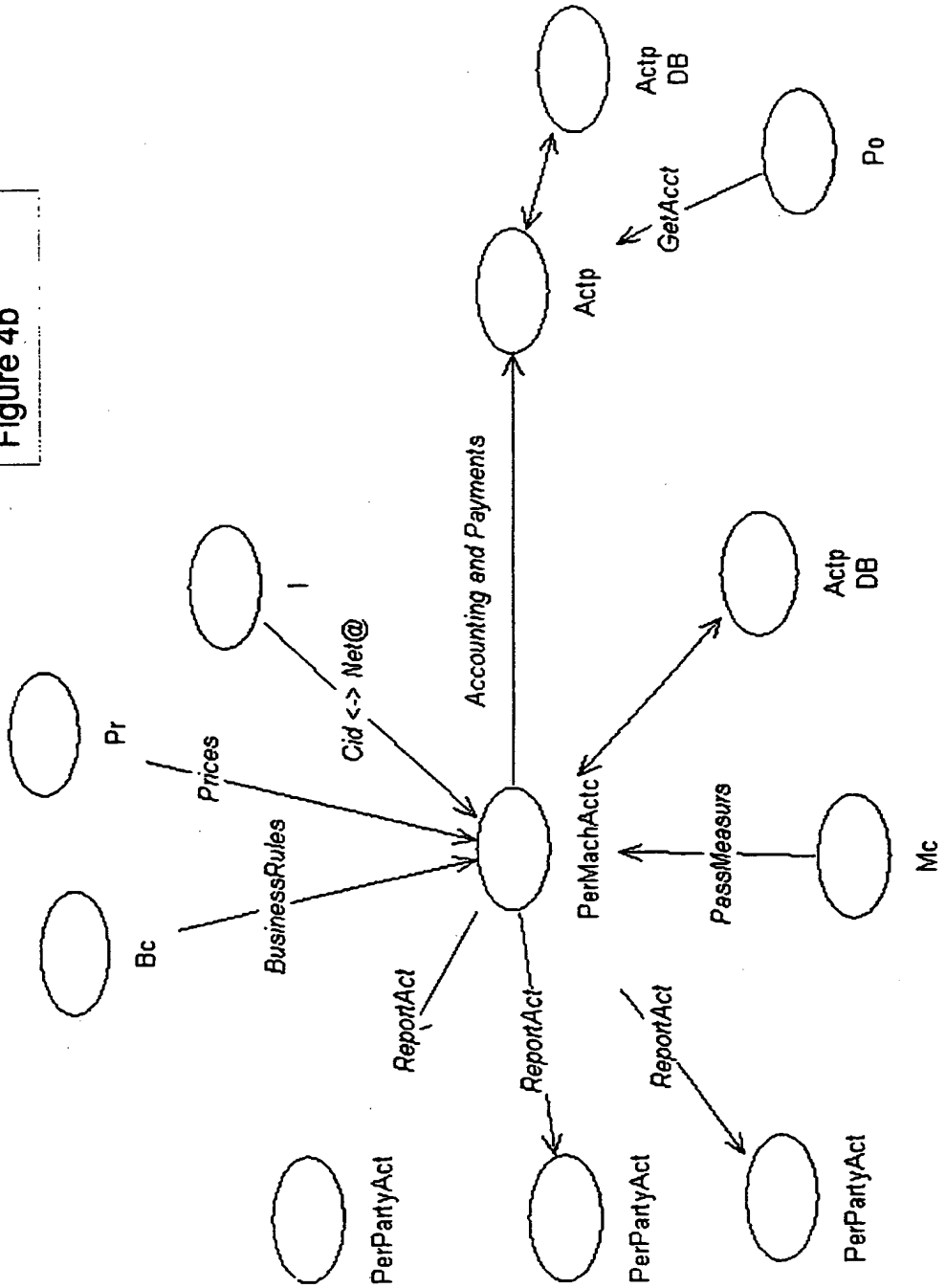
Actp DB
Po
GetAcct
Actp
Accounting and Payments
I
Cid <-> Net@
Pr
Prices
BusinessRules
Bc
ReportAct
ReportAct
ReportAct
PerMachActc
Actp DB
PassMeasurs
Mc
PerPartyAct
PerPartyAct
PerPartyAct

Figure 5a

Figure 5b

7/8

Figure 6

Figure 7

ABSTRACT

In a communications network, loading of network resources is detected locally at a customer terminal, and a tariff for network useage is varied automatically depending on the useage.